Remarks

Entry of the amendments, reconsideration of the application, as amended, and allowance of all pending claims are respectfully requested. Upon entry of the amendments, claims 1-65 are pending.

With the above amendments, applicants have provided the information for the related application, as well as changed the word "machine" in claims 45 and 58, as requested by the Examiner. Applicants have changed the word "machine" to "computing unit," which is a term used within applicants' specification (e.g., p. 7). Additionally, applicants have amended the independent claims to clarify that the moving of the queue from one processor to another processor results in the queue being resident in the processor to which it was moved and no longer resident in the other processor. Support for this amendment is inherent in the word "moving" which indicates that an object is changed from one place another place (see Webster's Ninth New Collegiate Dictionary), as opposed to "copying" which indicates that a duplicate of an object may be in two places at once. Support for the amendment may also be found in the specification itself which describes the process of moving the queue from one processor to another processor, in which it is then located on just the one processor (see FIGs. 3-6, as well as the accompanying text). Thus, no new matter is being added.

In the Office Action, dated May 13, 2004, claims 1-3, 9-17, 21-23, 29-37, 41-47 and 53-61 are rejected under 35 U.S.C. 102(b) as being anticipated by Blount et al. (U.S. Patent No. 5,222,217); and claims 4-8, 18-20, 24-28, 38-40, 48-52 and 62-64 are rejected under 35 U.S.C. 103(a) as being unpatentable over Blount in view of Dievendorff et al. (European Application 0280773). Applicants respectfully, but most strenuously, traverse these rejections for the reasons herein.

In one aspect, applicants' invention is directed to a processor taking over one or more queues of another processor of a communications environment. That is, for one reason or another, the queues of one processor are inaccessible, and thus, another processor takes over one or more of those queues. The queues being taken over are memory resident queues. That is, they are queues that reside in, for instance, local memory of a processor, as opposed to DASD or

some other storage unit. The queues are, therefore, volatile by nature. For example, if a processor is inactive, then the queues in memory are inaccessible and lost.

Previously, in order to access inaccessible memory resident queues, those queues would need to be rebuilt by the processor that owned those queues. However, in accordance with an aspect of the present invention, those queues are now taken over by at least one other processor by moving one or more of the queues to the other processor. The queues, once taken over, are then resident in the other processor's memory, and no longer resident in the first processor's memory.

In one particular example, applicants claim a method of switching queue ownership (e.g., independent claim 1). The method includes, for instance, obtaining an indication that a queue is to be taken over, the queue being resident in memory of a first processor; and moving the queue from the first processor to a second processor, the queue to be resident in memory of the second processor and not resident in memory of the first processor. Thus, in applicants' claimed invention, when a queue is moved from one processor to another, that queue is no longer resident in memory of the processor from which it is moved, but is now resident in memory of the processor to which it is moved. That is, the queue is only resident in one memory. This is very different from the teachings of Blount.

Blount teaches a system and a method for implementing operating system message queues with recoverable shared virtual storage. The system assures the reliability of system-wide shared data structures in the event of a failure of one of the processors by maintaining at least two copies of each data structure and by maintaining two copies of a table used in locating such data structures. Thus, Blount teaches a duplication procedure in which the queue is maintained on two processors and updates to the queue occur concurrently on the two different processors. This is very different from applicants' claimed invention.

In applicants' claimed invention, a queue is not duplicated on multiple processors, but instead, is moved from one processor to another processor. As a result of moving the queue, the queue is resident in memory of one processor. There is no such moving in Blount. Instead, in Blount, two copies of the data structure are maintained in at least two different processing units (see, e.g., Abstract). Blount teaches copying or duplicating, and not moving, as claimed by

applicants. Since the queue in Blount is already on a second processor, there is no need to move it to the second processor, as claimed by applicants.

Support for the rejection is indicated in Col. 8, lines 8-10 of Blount. These lines indicate that two copies of each page are kept in different processors which are only updated after the completion of a transaction which may modify such a page. The copies are updated simultaneously in the systems, and one processor is designated as the lead processor and another processor is designated as backup. Applicants respectfully disagree that these statements teach applicants' claimed invention of moving a queue from one processor to a second processor. Instead, applicants respectfully submit that those statements indicate that the queue is duplicated and maintained on two processors. Since the queue is duplicated, there is no necessity for moving the queue. That is, Blount would not require the moving of a queue from one processor to another processor, since the queue is concurrently maintained on two processors at the same time. Thus, Blount does not describe, teach or suggest applicants' claimed moving.

Further, applicants specifically indicate that the moving results in the queue being resident in memory of a second processor and not resident within the first processor, which is not described, taught, or suggested in Blount. As a matter of fact, Blount specifically teaches away from such a claimed element by requiring that the queue be simultaneously updated and maintained in two processors at the same time. Thus, applicants respectfully submit that Blount does not describe, teach or suggest applicants' claimed invention.

In addition to the above, applicants claim obtaining an indication that a queue is to be taken over, the queue being resident in memory of a first processor. It is indicated in the Office Action that this is taught by the fact that Blount teaches detecting a failure of a processor. Applicants respectfully submit that even if Blount taught the detecting of a failure of a processor that is not a teaching or suggestion that a queue is to be taken over. As described in applicants' Background, prior to this invention, if a memory resident queue of a processor became inaccessible, the queue would remain inaccessible until the processor became active again. There was no takeover procedure. Thus, a failure of a processor is not an indication that a queue is to be taken over. A reading of such a feature into Blount is merely hindsight reconstruction of

applicants' claimed invention. Blount does not describe, teach or suggest applicants' claimed element of obtaining an indication that a queue is to be taken over.

Based on the foregoing, applicants respectfully submit that Blount does not describe, teach or suggest one or more aspects of applicants' claimed invention. Therefore, applicants respectfully request an indication of allowability of independent claim 1, as well as the other independent claims.

The dependent claims are patentable for the same reasons as the independent claims, as well as for their own additional features. For example, newly added dependent claim 65 indicates that the moving is performed in response to an indication that the queue is to be taken over and not in response to performing a transaction in which the queue is processed. This is clearly not described, taught or suggested in Blount. Blount specifically teaches duplicating the queue in response to processing a transaction (see, e.g., Col. 8, lines 1-10). That is, as the queue is updated in one processor, it is updated in another processor, such that duplication of the queue is maintained. In contrast, in applicants' claimed invention, the queue is not moved from the memory of one processor to the memory of another processor unless there is an indication that the queue is to be taken over. There is no moving for normal processing. Thus, applicants respectfully submit that dependent claim 65 is patentable over Blount.

Dievendorff does not overcome the deficiencies of Blount. In particular, Dievendorff fails to teach or suggest one or more aspects of applicants' claimed invention. Moreover, applicants respectfully submit that the combination of Blount and Dievendorff is improper. One skilled in the art would not look to combine a reference that teaches duplication of data structures with another reference that teaches recovery using checkpoints and logs. The duplication eliminates the need for checkpoints and logs. Therefore, the combination is improper.

Based on the foregoing, applicants respectfully submit that applicants' claimed invention is patentable over Blount and Dievendorff, either alone or in combination. Therefore, applicants respectfully request an indication of allowability for all pending claims.

Should the Examiner wish to discuss this case with applicants' attorney, please contact applicants' attorney at the below listed number.

Respectfully submitted,

Blanche E. Schiller

Blanche E. Schiller
Attorney for Applicants
Registration No.: 35,670

Dated: August 12, 2004.

HESLIN ROTHENBERG FARLEY & MESITI P.C.
5 Columbia Circle
Albany, New York 12203-5160
Telephone: (518) 452-5600
Facsimile: (518) 452-5579